C. Remarks

The presently pending claims stand rejected variously based on the newly cited reference Primak et al (US Publication 2002/0010783) and the prior cited references Ballard (US Patent 6,078,960), Mangipudi et al (US Patent Publication 2004/0162901), and Kawata (US Publication 2002/0032777). Applicants respectfully assert that the cited art, as cited in combination, fails to teach or suggest the presently claimed invention.

The teachings of Ballard and Mangipudi were summarized in the prior Response.

Summary Analysis of Primak:

The Primak reference describes a client/server system that implements a load-balancing algorithm performed entirely by and between the server systems.

Each of the server systems includes a load-balancing module that determines whether and to which other specific server to redirect an inbound client request. Inbound client requests directed to the server site are multicast by a front-end router to the individual servers. A hash function is separately and identically performed on all of the servers to resolve the one server that will consider the client request. That one server (call it Server A for convenience of discussion here) then selects, by itself, another server (Server B) to substantively respond to the client request. The Server A returns an ordinary HTTP redirection message to the client, causing the client to automatically reissue the client request to the redirection target, Server B. As redirection messages are commonplace and provide no explanation of the cause or purpose of redirection, the client is entirely unaware that its request is being redirected to effect load balancing; the client is entirely unaware that it is even incidentally participating in a load-balancing algorithm.

As part of determining the redirection target server, the Server A does consider the content type of the client request. Rather than being generic, the server sytems are taught as subsetted based on content type (*.cgi, *.html, etc). The Server A selects, by itself, the Server B from among those servers known by Server A as belonging to the appropriate content-defined subset of of servers. From the view-point of a client, the client merely presents the server site with a request inherently defined by a content type. On receipt of a redirection message, the client merely re-presents its same request to the redirection target server. The client is provided with no information to know that it is being redirected to

implement load-balancing; the reference provides no teaching or suggestion that the client retain any information about the redirection for any purpose whatsoever. Rather, the explicit teaching of Primak is that (1) all initial inbound client requests are generically directed to the server site and not to any individual server, and (2) all load-balancing occurs without any active participation of any client system, *i.e.*, entirely transparent to the clients.

Summary Analysis of Kawata:

The Kawata reference describes a load-balancing system requiring, as an essential element, a load balancing router (100) positioned in front of a set of servers (¶40). All client requests are received by and rerouted by the load balancing router (100) directly to the specific server determined by the router to handle the request. The server response return path is through the load balancing router (100) as well. Consequently, the performance of the load-balancing operation, as well as the identity of the particular server responding to a client request is entirely hidden from the clients.

Load balancing is predicated on accumulating work load estimates for outstanding requests forwarded to each of the servers fronted by the load balancing router (100). The work load for a particular request is estimated from pre-computed and stored test data correlated to different features of a client request and the server chosen to service the request.

The test data is generated in advance under isolated test conditions for each server fronted by the load balancing router (100) (¶74-76). As explicitly taught, a test machine (1100) is used to create an artificial load on a specific server to collect raw response time data (¶74). The server-under-test accumulates raw CPU loading data that is subsequently supplied to the test machine (1100) (¶74). This raw data is compiled to test data tables by the test machine (1100) and then forwarded (¶75) to the load balancing router (100) for final organization and storage in local tables (¶76). Throughout the test data generation, the load balancing router (100) operates merely as a conventional router.

At no point does Kawata teach or suggest that the load balancing router (100) communicates work load estimates to any client or even to any server. Instead, the load balancing router (100) of Kawata is the sole arbiter of load-balancing decisions. Consequently, Kawata does not teach or suggest the distribution of estimated or real work

load information between clients and servers let alone in execution of a load-balancing algorithm. Kawata simply does not contemplate any need for cooperation between the disclosed load balancing router (100) and any clients and servers to implement load-balancing decisions.

Summary Analysis of the Uncited Prior Art Mentioned in Kawata:

Kawata provides an exceedingly brief summary of two load-balancing methods considered by Kawata to be prior art. Since the art itself is uncited, only the literal summaries presented in Kawata, and not the underlying text, can be considered in determining the patentability of the pending claims (M.P.E.P. §706).

The first summary describes a set of servers where:

> ... each server periodically measures the number of IP packets per unit time and informs a state management server of its own load status. The client looks at the load status for each server in the state management server and sends its service request to the server with the lowest load.

Lacking any express or suggested reason to believe otherwise, a person of ordinary skill in the art would consider the stated "measured number of IP packets" to be of anonymous IP packets. That is, the IP packets are counted without regard of whether the packet traffic is from a client or another server and certainly not with regard to any particular client.

Further, a person of ordinary skill would consider the "load status" reported to the "state management server" to be likely nothing more than a single value sufficient to allow comparison. As taught by Kawata, different servers can have different absolute processing capabilities. In order for an arbitrary server to report an independently comparable "load status," the value reported by a server must be a pre-calculated relative loading value. Otherwise, the clients could not identify the "server with the lowest load" merely from the "load status for each server." Consequently, the clients base their server selection on nothing more than normalized values that do not reflect loading with respect to any particular client.

No other credible explanation of the load-balancing operation can be drawn from the first summary.

The second summary describes a TCP connection router that operates as:

> ... a load balancer [] interposed between multiple clients and multiple servers. The load balancer and each of the servers periodically measure load evaluation values for the servers, and servers to which requests are to be sent are determined dynamically from these load estimation [sic] values.

A person of ordinary skill would readily recognize this "load balancer" as implementing the same fundamental architecture as Kawata. The evident difference being that, rather than using estimated loading values, actual loading values are collected at predefined periodic intervals from the servers themselves. This summary is explicit in stating that the sole determinant for routing is the load evaluation values retrieved from the servers. Thus, there is no credible basis to believe that a person of ordinary skill would read the summary as teaching or suggesting any meaningful participation of clients in the load-balancing operation described.

Rejection of Claims 1 - 6 in view of Kawata and Primak:

Independent Claim 1 requires:

> A method of cooperatively load-balancing a cluster of server computer systems for servicing client requests issued from a plurality of client computer systems ...
>
> a) selecting, by a client computer system, a target server computer system ... using available accumulated selection basis data ...
>
> b) evaluating, by said target server computer system, said particular client request to responsively provide instance selection basis data dynamically dependent on the configuration of said target server computer and said particular client request; and
>
> c) incorporating said instance selection basis data into said available accumulated selection basis data ... . (Emphasis added.)

The claim thus requires an active interoperation – "cooperatively load-balancing" – between a plurality of client systems relative to a separate plurality of server systems. The interoperation is specified as the dynamic generation of instance selection basis data by the

server dependent on both the "configuration" of the target server and the "particular client request."

The claim further requires that the selection of the target server computer system be made by a client computer system using available accumulated selection basis data that incorporat[es] said instance selection basis data.

Consequently, the "cooperatively load-balancing" is achieved by a specific client selecting a target server using selection basis data accumulated from instance basis data generated by the target server based on specific client requests.

Kawata clearly fails to teach or suggest any involvement of a client selecting a particular target server. Likewise, Primak only teaches or suggests clients that are entirely unaware of the implementation of any load-balancing operation.

The summary of the first uncited reference fails to teach or suggest that a client accumulate any information. This summary certainly fails to teach or suggest that instance load data be generated by a specific target server with regard to a specific client request and further be provided to and accumulated by a client. The summary also fails to teach or suggest that a particular client actively operates to select a specific target server based on load information accumulated by that client specific to that target server.

Consequently, none of the references teach or suggest the dynamic generation and contribution of instance load data from specific servers to specific clients for use by those clients in selecting a specific server to service a request. The references thus do not teach or suggest the present invention as set forth in Claim 1. Reconsideration of the rejection of Claim 1, including for the same reasons dependent Claims 2-6, is respectfully requested.

Rejection of Claims 7 - 19, 25 - 27, and 29 - 30 in view of Mangipudi and Kawata:

Independent Claim 7 requires:

> A method of load-balancing a cluster of server computer systems in the cooperative providing of a network service to host computers operating mutually independent of one another ...

a) selecting, independently by each of a plurality of host computers, server computers within a computer cluster ...

c) receiving, in regard to said respective service requests ... load and weight information from respective said server computers, wherein load and weight information is dynamically generated by respective said server computers; and

d) evaluating, by each of said plurality of host computers, respective load and weight information ... as a basis for a subsequent performance of said step of selecting. (Emphasis added.)

Claim 7 requires the provision of both "load and weight information" from the servers to a requesting "host computer," *i.e.*, client, in response to a "respective service request." This "load and weight information" is required by Claim 7 to be "dynamically generated by respective said server computers." Claim 7 is further specific that the "load and weight information" be received by the host computers that originate the "respective service requests."

As established in the prior Response, the CPU system load values retrieved by Mangipudi from the servers corresponds to only the "load" information required to be returned by the present claim. Mangipudi does not teach or suggest the retrieval of any other information, *i.e.*, "weight information," information from the servers and certainly not "weight information ... dynamically generated ... by said server computers." Furthermore, there is no suggestion presented in Mangipudi that any other information relevant to load-balancing even exists on or can be retrieved from the servers.

Kawata is now asserted as showing "the receiving of weight information from the server through load-balancing in selecting a particular server." Kawata, however, is explicit that the disclosed weighting values are maintained internal to the load-balancing router (¶55) and are used only internal to the load-balancing router to make routing decisions (¶59). Furthermore, the Kawata weighting values are artificially generated estimate values generated in a test setup by a test machine (¶74). The information from which the weighting values are generated is supplied from the test machine and not from the servers (¶75). The load-balancing router appears to internally compute the estimate values (¶76), which are thereafter used only internal to the load-balancing router.

Consequently, the weighting values of Kawata are clearly not the "weight information ... dynamically generated by ... said server computers" that is received by the "plurality of host computers [that] issue respective service requests." The use of artificially generated load information by an external test machine later processed into load estimate weightings likewise fails to suggest the use of distinct "load and weight information" by the "plurality of host computers" to independently select server computers for receipt of service requests.

Consequently, the combination of Mangipudi and Kawata does not teach or suggest the retrieval and use of both "load and weight information" from a server for use by one of a plurality of host computers in selecting, to effectuate cooperative load-balancing, a particular server to receive a particular service request.

Accordingly, Applicants respectfully request reconsideration of the rejection of Claim 7 as obvious in view of Mangipudi and Kawata. Reconsideration of the rejection of Claims 7-12, for at least the same reasons presented in regard to Claim 7, is also requested.

Similar to Claim 7, independent Claim 13 requires:

> a) a plurality of server computers individually responsive to service requests ..., wherein said server computers are operative to initially respond to said service requests to provide load and weight values, wherein said load and weight values represent a current operating load and a policy-based priority level of a respective server computer relative to a particular service request; and
>
> b) a host computer system operative to autonomously issue said service requests [... and ...] to select a target server computer ... to receive an instance of said particular service request based on said load and weight values. (Emphasis added.)

For the reasons advanced above with respect to Claim 7, Applicants respectfully assert that Claim 13 and dependent Claims 14 - 19 are not obvious in view of Mangipudi and Kawata. The claim requires "load and weight values" to be provided by a server computer to a host computer for use in "selecting a target server computer [to receive a] particular service request." Accordingly, Applicants respectfully request reconsideration of the rejection of Claims 13 - 19.

Independent Claim 25 requires:

> c) receiving from said particular server computer system with respect to said particular client request instance selection qualification information discretely determined by said particular server computer system dynamically with respect to said particular client request, wherein said instance selection qualification information including a load value reflective of the current performance capability of said particular server computer system and a weight value reflective of the anticipated performance capability of said particular server computer system with respect to said particular client request, wherein said instance selection qualification information is incorporated into said accumulated selection qualification information. (Emphasis added.)

Similar to Claim 1, Claim 25 requires a load-balancing performed cooperatively by the clients and servers. This cooperative relation is archived by the servers evaluating "particular" client requests and returning to the clients "instance selection qualification information" that is specific to the "particular client request." This "instance selection qualification information" is accumulated and used by the client as the basis for selection of a particular server computer system for receipt of a particular client request.

Similar to Claim 7, Claim 25 further requires the server computer systems to return both load and weight information to the clients: said instance selection qualification information including a load value reflective of the current performance capability of said particular server computer system and a weight value reflective of the anticipated performance capability of said particular server computer system with respect to said particular client request.

Consequently, Mangipudi and Kawata fail to teach or suggest the present invention as set forth in Claim 25. Reconsideration of the rejection of Claim 25, including for the same reasons dependent Claims 26-27, is respectfully requested.

Dependent Claim 29, as amended, specifies that:

> said weight value part of said instance selection qualification information includes a relative prioritization of said particular client request with respect to said particular server computer system.

As established above, neither Mangipudi nor Kawata teaches or suggests the dynamic determination of "a weight value reflective of the anticipated performance capability of said particular server computer system with respect to said particular client request," as set forth in independent Claim 25. Claim 29 qualifies the weight value as including a "relative prioritization," considering the "particular client request" in specific correspondence with the "particular server computer system."

The load-balancing performed by Mangipudi is dependent on only load values from the servers. The Mangipudi servers do not provide any information dynamically determined by a server that represents any "relative prioritization" of a "particular client request" with respect to a "particular server computer system."

Consequently, Claim 29 and Claim 30, as dependent therefrom, are not obvious in view of the combined teachings of Mangipudi and Kawata. Accordingly, Applicants respectfully request reconsideration of the rejection of Claims 29 and 30.

Rejection of Claims 20, 22-24, 31, and 33-36 in view of Mangipudi, Ballard, and Kawata:

Independent Claim 20 requires:

> a) ... a server computer of said first plurality provides a response, including dynamically determined load and weight information, in acknowledgment of a predetermined service request issued to said server computer system ...
> b) ... wherein said client computer system is reactive to said response ... and wherein said client computer system is responsive to said load and weight information of said response in subsequently autonomously selecting said first and second server computer systems.

Independent Claim 31 requires:

> c) second processing said particular client request ... by said particular target server system to dynamically generate instance selection information including a load value for said particular target server system and reflective of a combination of said particular client request and said particular target server system and a relative weighting value reflective of the combination of said particular client request and said particular target server system; and
>
> d) incorporating said instance selection information into said accumulated selection information for subsequent use in said step of selecting,
>
> wherein said step of selecting matches said particular client request, including said attribute data, against corresponding data of said accumulated selection information to choose said particular target server system based on a best corresponding combination of relative weighting value and load value.

As established in the prior Response, neither the Mangipudi nor Ballard reference teaches or suggests the server production of both "load and weight information" or, indeed, any server "dynamically determined" information "in acknowledgment of a predetermined service request" as required by Claim 20. As further established above, the Kawata reference only teaches the use of estimate weightings and then only internal to a dedicated load balancing router. Furthermore, the Kawata weightings are received only from a test machine operating under test conditions where no load-balancing is actually being performed.

Nothing in the combination of the references serves to suggest, let alone motivate, a person of ordinary skill the art to consider having any server contribute dynamic preference information specific to particular service requests for use in a load-balancing algorithm performed among and with the active participation of a first plurality of server computers and a second plurality of client computers, where a client computer operates to "autonomously select a first server computer ... to which to issue said predetermined service request [based on] said load and weight information."

Particularly in regard to Claim 31, the load-balancer use of the "dynamically generate[d] instance selection information" is expressly required. When considering the "accumulated selection information," the "particular client request" is matched to the selection information to find a most preferred server for the client request based on "a best corresponding combination of relative weighting value and load value." Such a consideration of the server generated preference information, including both load and weighting information, is nowhere taught or suggested by the cited references, either alone or in combination.

Accordingly, Applicants respectfully request reconsideration of the rejection of Claims 20 and 31 as obvious in view of Mangipudi, Ballard and Kawata. Reconsideration of the rejection of Claims 22-24 and 33-36, for at least the same reasons presented in regard to Claims 20 and 31, is also requested.

Conclusion:

In view of the above Amendments and Remarks, Applicants respectfully assert that Claims 1 – 20, 22 – 27, 29– 31, and 33 – 36 are now properly in condition for allowance. The Examiner is respectfully requested to take action consistent therewith and pass this application on to issuance. The Examiner is respectfully requested to contact the Applicants' Attorney, at the telephone number provided below, in regard to any matter that the Examiner may identify that might be resolved through a teleconference with the Examiner.

Respectfully submitted,

Date: 11/6/2006

By: Gerald B. Rosenberg

Gerald B. Rosenberg
Reg. No. 30,320

NEWTECHLAW
285 Hamilton Avenue, Suite 520
Palo Alto, California 94301
Telephone: 650.325.2100